
Universidade de Aveiro
2009
Departamento de Electrónica, Telecomunicações e Informática

Ricardo João Luís Marques Correia

Scrambler para VoIP

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática
2009

Ricardo João Luís Marques Correia

Scrambler para VoIP

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Dr. Francisco António Cardoso Vaz, Professor Catedrático (Aposentado) do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro e do Dr. Paulo Jorge dos Santos Gonçalves Ferreira, Professor Catedrático do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Dedico este trabalho à minha família e aos meus amigos pelo incansável apoio.

o júri

presidente

Prof. Dr. Ana Maria Perfeito Tomé
Professora Associada do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

vogais

Prof. Dr. Francisco António Cardoso Vaz
Professor Catedrático (Aposentado) do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro (orientador)

Prof. Dr. Paulo Jorge dos Santos Gonçalves Ferreira
Professor Catedrático do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro (co-orientador)

Prof. Dr. Manuel Alberto Pereira Ricardo
Professor Associado da Faculdade de Engenharia da Universidade do Porto

agradecimentos

Agradeço a todos os meus amigos que me acompanharam ao longo destes anos. Muitos já me acompanhavam antes de integrar no curso outros fui felizmente conhecendo ao longo dele. Todos me ajudaram para que esta fase da minha vida fosse superada.

À minha família que sempre me apoiou incondicionalmente em todas as decisões que tomei.

Ao Professor Dr. Francisco Vaz pelo apoio e disponibilidade incondicional que sempre teve para resolver todos os problemas que surgiram ao longo do trabalho.

Ao Professor Dr. Paulo Ferreira pelo apoio que sempre prestou quando necessário.

palavras-chave Confidencialidade*, scrambling, Diffie-Hellman*, codecs

resumo A necessidade de se preservar a confidencialidade numa conversa telefónica é um facto dos nossos dias.
Pretende-se com este trabalho dar uma resposta a este problema propondo um sistema original de *scrambling* do sinal no domínio das frequências. O sistema inclui uma troca de chaves públicas que geram uma chave secreta comum entre o emissor e o receptor, baseado no método de *Diffie-Hellman*. Além da implementação apresentam-se resultados de testes efectuados sobre o sistema de *scrambling* proposto associado a vários codecs de uso comum.

keywords Confidentiality*, scrambling, Diffie-Hellman*, codecs

abstract The need to preserve confidentiality in a telephone conversation is, nowadays, a motive of concern for most of us. This work pretends to address this issue by proposing an original signal *scrambling* system in the frequency domain. The system includes a public key exchange which generates a common secret key, based on the *Diffie-Hellman* method, at the transmitter and the receiver.
The implementation and some results of the proposed solution using common conventional codecs are presented.

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1 Introdução

A comunicação é a transmissão de informação entre um emissor e um receptor através de um canal de comunicação. Desde sempre se procurou que, em certas condições, houvesse confidencialidade na comunicação no sentido em que apenas o emissor e receptor tivessem acesso à informação transmitida. Esta confidencialidade era garantida ou por utilização de um canal de transmissão seguro (isto é, não acessível a terceiros) ou garantir que o conteúdo da mensagem não era compreensível por terceiros. Este último porém é o que interessa ao presente trabalho em que se vai tratar de garantir a confidencialidade numa comunicação telefónica. Contudo, com a evolução do tempo e da tecnologia, torna-se cada vez mais difícil confiar na privacidade/confidencialidade das chamadas telefónicas.

O canal telefónico começou por ser analógico, isto é, o sinal de voz era convertido num sinal eléctrico num microfone e era transmitido como sinal eléctrico para o receptor onde o altifalante o convertia novamente em onda sonora.

A confidencialidade é facilmente violada pois a tecnologia permite a escuta do canal telefónico com toda a fiabilidade e a mensagem pode ser facilmente escutada.

Para garantir a confidencialidade surgiu a ideia de *scrambling*: o sinal era continuamente modificado (normalmente recorrendo a técnicas de modulação) de uma forma apenas conhecida pelo emissor e receptor.

(Embora a palavra inglesa *scramble* seja traduzível por "baralhar" ou "misturar", a nossa opção foi manter o inglês para esta palavra ou suas derivadas).

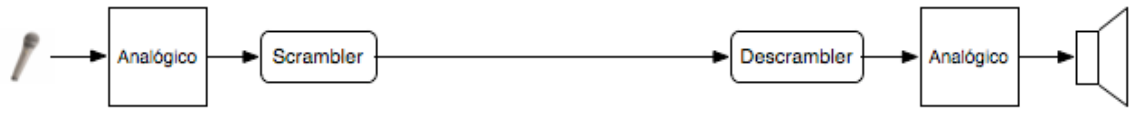

Figura 1 - *Scrambling* analógico

A tecnologia foi evoluindo e passou-se a utilizar sinais digitais. Nesta situação o sinal eléctrico vindo do microfone é digitalizado na ADC (*Analog-to-Digital Converter*) e transmitido digitalmente. No receptor o sinal digital é reconvertido em analógico pela DAC (*Digital-to-Analog Converter*) e depois convertido em onda sonora.

O *scrambling* é facilmente incorporado como se indica na figura seguinte.

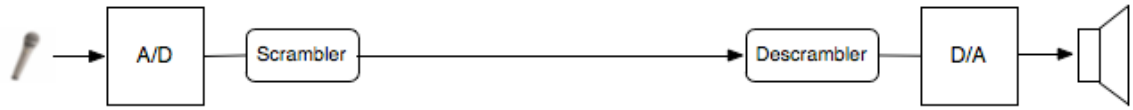

Figura 2 - *Scrambling* digital

A tecnologia digital abre grandes hipóteses pois as amostras na sua forma digital podem ser modificadas de múltiplas maneiras.

Assim por exemplo tem-se o *scrambler* de multiplicação:

Este tipo de *scrambler* faz uma multiplicação do sinal de entrada discreto por uma função de transferência de um sistema linear invariante no tempo. O *descrambler* é semelhante, apenas se distinguindo por não ser recursivo. Tem sincronização própria, isto é, não precisa de sincronização do frame.

Nas figuras seguintes apresenta-se esquematicamente um exemplo do *scrambler/descrambler* de multiplicação.

O *scrambler* é definido por uma função de transferência polinomial que neste caso é $1/(1-z^{18}-z^{23})$. O *descrambler* é definido por $1-z^{18}-z^{23}$ e é trivial verificar que a saída do *descrambler* é igual à entrada do *scrambler*.

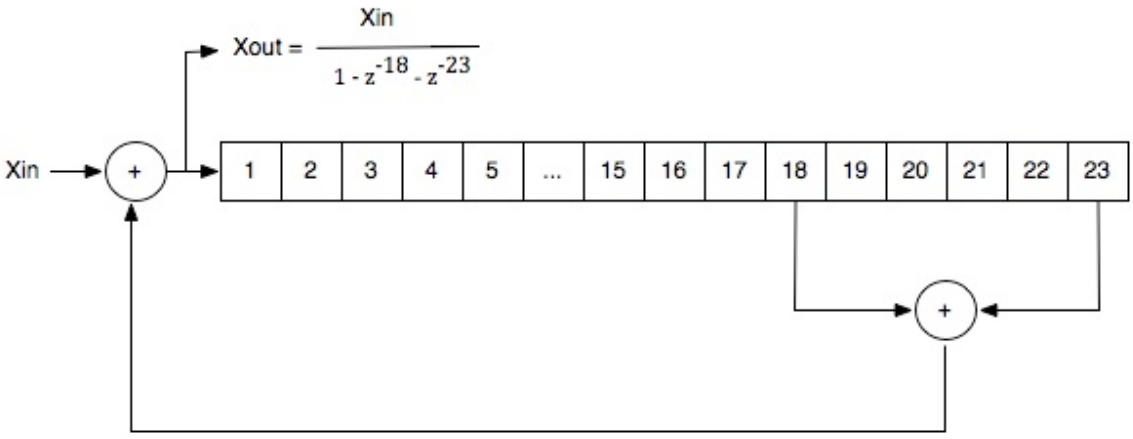

Figura 3 - *Scrambler* de multiplicação

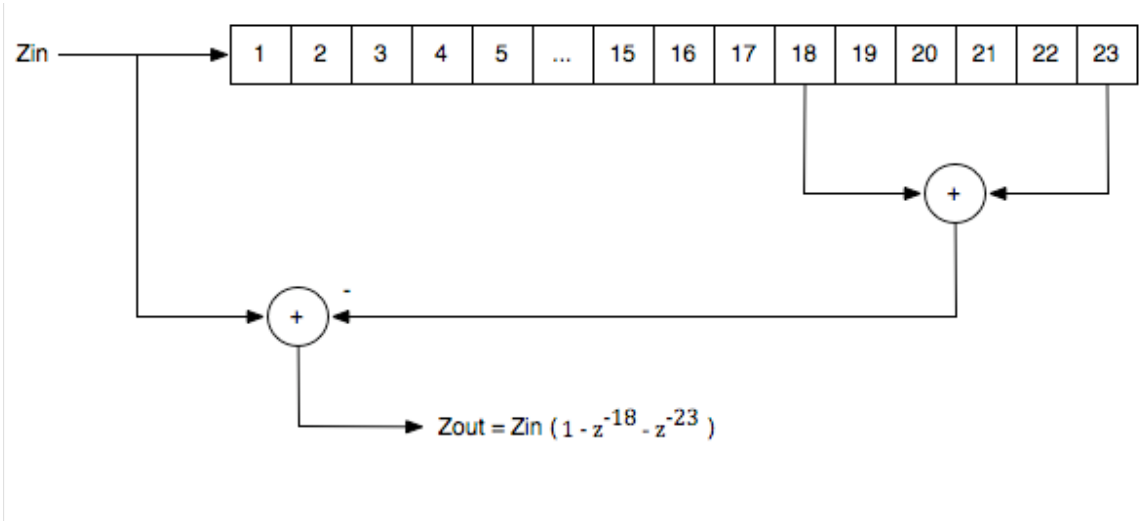

Figura 4 - *Descrambler* de multiplicação

Mais recentemente, todo o processo de comunicação passou a ser digital e entre o emissor e o receptor passou a haver uma rede que se pode considerar como uma entidade que garante a transmissão da informação com fiabilidade, mas em geral, sem confidencialidade.

Uma das redes usada na actualidade é a rede do tipo IP, significando que cada utilizador é caracterizado por um endereço (IP *address*) e que a informação é transmitida obedecendo a certo tipo de formato. A comunicação do sinal digital e a rede é feita através de dispositivos que se chamam codec (codificador/descodificador).

A confidencialidade pode ser garantida por encriptação directa na rede e nesse caso estará sempre dependente do proprietário da rede (operador).

Pode-se no entanto pensar numa solução alternativa e não dependente da rede, a utilização de *scrambling* do sinal digital feito antes do codec e é essa a proposta que se irá explorar no presente trabalho.

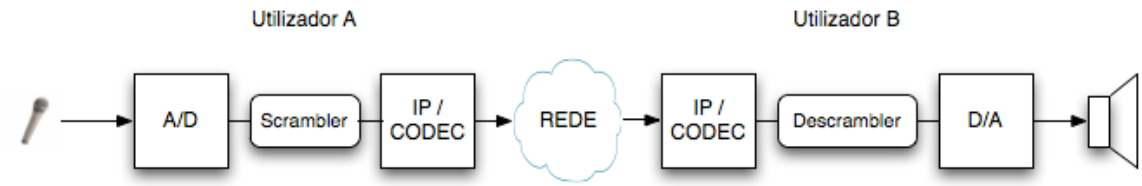

Figura 5 - *Scrambler* digital com codec

1.1 Objectivos

O objectivo principal deste trabalho consiste em demonstrar a viabilidade de um sistema que permita encriptar a voz usando um método de chave pública/privada sobre um canal de comunicação *VoIP* (*voice over internet protocol*).

Pretende-se também avaliar o comportamento do sistema perante diferentes codecs.

Pretende-se ainda avaliar a sua robustez perante ruído introduzido pelo canal de transmissão.

A implementação do sistema é feita usando o programa Matlab.

Capítulo 2 Estado de arte

Este trabalho baseia-se em três conceitos fundamentais. São eles o *scramble*, criptografia assimétrica de chave pública e *VoIP*.

2.1 *Scramble*

O primeiro conceito *scramble* consiste num dispositivo que transpõe ou inverte sinais, ou por outro lado, codifica a mensagem no transmissor para a tornar ininteligível no receptor que não está equipado com o aparelho de descodificação apropriado. É realizado através da adição de componentes para o sinal original ou a mudança de algum componente importante do sinal original a fim de tornar a extracção do sinal original difícil.

O método mais simples de *scramble* de voz consiste em dividir a informação da voz em duas bandas e invertê-las em torno da frequência da portadora, fazendo com que os baixos tons soem como altos e vice versa. Esta técnica é conhecida como *Split Band Voice Inversion* (*VSB*). Outra técnica que existe é *Rolling Code Voice Inversion* que usa um código aleatório para escolher a frequência da portadora e depois muda esse código em tempo real. Juntando estas duas técnicas anteriores consegue-se mais uma que se dá o nome de *Split Band Variable Inversion*.

O *scrambling* digital pode ser feito por alteração da ordem de transmissão das amostras (amostra a amostra ou por blocos) ou alterando as amostras como no exemplo apresentado (*scrambler* de multiplicação).

Na figura em baixo mostra-se um exemplo de *scrambler* que consiste num reordenamento aleatório das amostras. Os dois gráficos de cima correspondem ao sinal original, sendo um gráfico do sinal e o outro o seu espectro na frequência (calculado através da transformada de Fourier). Os dois gráficos de baixo referem-se ao sinal transformado pelo *scrambler*.

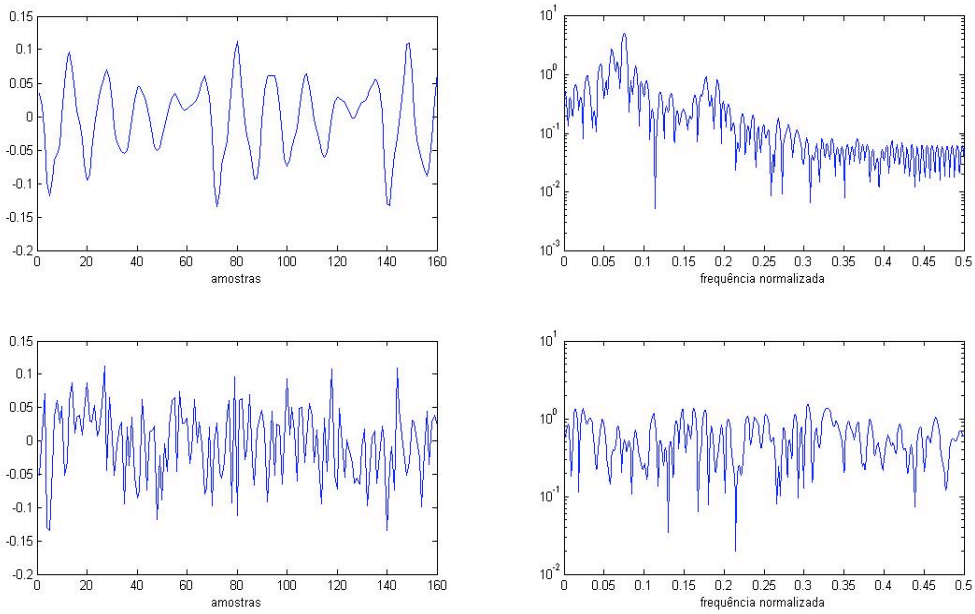

Figura 6 - Exemplo de *scrambler*

A observação deste gráfico mostra que não só no tempo o sinal é modificado como também o é na frequência e isso constitui em geral um problema quando se vão utilizar os codecs.

Actualmente a utilização de *scramblers* não é muito difundida, pois o uso de técnicas de encriptação da informação digital é em geral considerada suficiente. A nossa proposta de usar *scramblers* destina-se a aplicações em que se pretende estabelecer uma ligação segura entre dois utilizadores, fácil de implementar e independente dos operadores de rede.

2.2 Encriptação

A encriptação é o estudo dos princípios e técnicas pelas quais a informação pode ser transformada da sua forma original para outra ilegível, de forma que possa ser conhecida apenas por seu destinatário (detentor da chave privada). Nos últimos tempos a necessidade de proteger informação, de modo que alguém indesejável não tenha acesso ao seu conteúdo, tem sido imperiosa. Um dos métodos para se criar essa protecção é a criptografia. O uso corrente da criptografia é encontrado, por exemplo, em transacções bancárias via Internet ou em compras on-line com cartão de crédito. Assim a criptografia torna-se um agente de segurança num sistema de comunicações.

Num algoritmo de criptografia assimétrica, uma mensagem cifrada com uma chave pública apenas pode ser decifrada pela chave privada correspondente. Estes algoritmos podem ser utilizados para autenticidade e confidencialidade. Para

confidencialidade, a chave pública é usada para cifrar mensagens, assim apenas o detentor da chave privada pode decifrá-la. Para autenticidade, a chave privada é usada para cifrar mensagens e assim garante-se que apenas o detentor da chave privada cifrou a mensagem que qualquer pessoa pode decifrar com a chave pública.

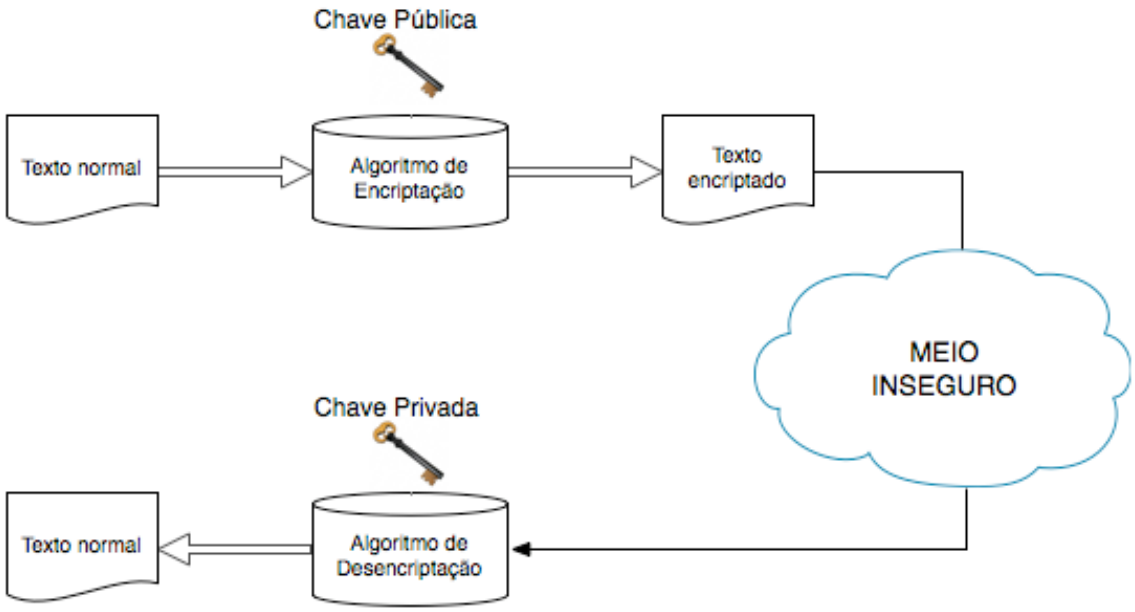

Figura 7 - Criptografia assimétrica

Dois exemplos de criptografia assimétrica são os algoritmos de *RSA* (*Ron Rivest, Adi Shamir e Leonard Adleman*) e *Diffie-Hellman* (*Whitfield Diffie e Martin Hellman*). Certos conceitos básicos de teoria de números são necessários para a compreensão do algoritmo de *RSA*.

2.3 Notas sobre a teoria dos números

Uma das ferramentas mais importantes na teoria dos números é a aritmética modular, que envolve o conceito de congruência. Uma congruência é a relação entre dois números que, divididos por um terceiro - chamado módulo de congruência - deixam o mesmo resto.

Exemplo:

Supõe-se que *a*, *b* e *m* são números inteiros diferentes de zero. Diz-se que *a* é congruente de *b* módulo *m* se *m* dividir *a* e *b*. Usa-se a notação seguinte:

$$a \equiv b (\mathrm{mod}\, m)$$

Números Primos relativos: o seu único factor comum é 1.

A adição e multiplicação modular *n* satisfazem quase todas as propriedades das operações aritméticas normais com uma excepção. A equação

$$ab \equiv ac \pmod{n}$$

não implica que $b \equiv c$, apenas implica quando *a* e *n* são primos relativos.

Teorema de Fermat **:** Se *p* é um número primo e *a* não é divisível por *p*, então

$$a^{p-1} \equiv 1 \pmod{p}$$

Demonstração: $ax \bmod p$, para x = 0,1,...,$p - 1$ define uma permutação em {0,1,...,p - 1}. Classificando a sequência de $p - 1$ elementos

$$a \bmod p,\ 2a \bmod p,\ldots,\ (p-1)a \bmod p$$

leva a {1,2,..., p - 1} (zero é excluído porque a0 = 0).

Multiplicando os números leva a

$$a\ 2a\ 3a \ldots (p-1)a = (p-1)!a^{p-1}$$

mas também a

$$1\ 2\ 3 \ldots (p-1) = (p-1)!$$

Então,

$$(p-1)!a^{p-1} \equiv (p-1)! \bmod p,$$

e como (p - 1)! é primo relativo de p, o factorial pode ser eliminado, concluindo-se assim

$$a^{p-1} \equiv 1 \pmod{p}.$$

Função totiente de Euler**:** $\varphi(n)$, o número de inteiros positivos menor que *n* e primo relativo de *n*.

Obviamente, $\varphi(p) = p - 1$ para qualquer número primo *p.*

Sendo $n=pq$, sendo *p* e *q* números primos. Então,

$$\varphi(n) = (p-1)(q-1) = \varphi(p)\,\varphi(p).$$

Demonstração: 0,{p,$2p$,...,$(q-1)p$}, e {q,$2q$,...,$(p-1)q$} não são primos relativos de *n*. Então,

$$\varphi(n) = n - [1 + (q-1) + (p-1)] = pq + 1 - q - p = (p-1)(q-1).$$

Teorema de Euler**:** Se *a* e *n* são primos relativos,

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Demonstração: Quando *n* é primo, $\varphi(n) = p - 1$, o resultado segue pelo teorema de Fermat. Quando *n* não é primo, seja

$$S = \{s_1, s_2, \ldots, s_{\varphi(n)}\}$$

o conjunto de $\varphi(n)$ números inteiros primos relativos de *n*. Multiplicando cada um dos inteiros por *a*, módulo n:

$$P = \{as_1 \bmod n, as_2 \bmod n, \ldots, a_{\varphi(n)} \bmod n\}$$

Cada um dos elementos as_i é primo relativo de *n* (s_i e *n* são também primos relativos). Em consequência, P e S contêm os mesmos elementos, ordenados de maneira diferente.

Multiplicando os elementos de S leva a $\prod_{i=1}^{\varphi\{n\}} S_i$

Multiplicando os elementos de P juntos leva a $\prod_{i=1}^{\varphi\{n\}} aS_i \bmod n$

Finalmente, como P e S estão relacinados por uma permutação,

$$\prod_{i=1}^{\varphi\{n\}} S_i \equiv a^{\varphi(n)} \prod_{i=1}^{\varphi\{n\}} S_i \ (mod\ n)$$

que significa

$$a^{\varphi(n)} \equiv 1 \ (mod\ n)$$

2.4 Algoritmos

2.4.1 *RSA*

Um dos algoritmos mais seguros de encriptação de informações actuais originou-se dos estudos de *Ronald Rivest*, *Adi Shamir* e *Leonard Adleman*.

O princípio do algoritmo é construir chaves públicas e privadas utilizando números primos.

A ideia é encriptar blocos de bits de tamanho $2^K < n$ usando

$$C = P^e \bmod n ,$$

onde *e* e *n* são números inteiros conhecidos (chave pública). A desencriptação é baseada em

$$P = C^d \bmod n ,$$

onde *d* é secreto. Assim conclui-se que se deverá ter

$$P^{ed} = P \bmod n ,$$

para todo $P < n$.

A possibilidade de encontrar *e* e *d* é garantida pelo teorema de Euler. Assim :

Para gerar as duas chaves, escolhem-se dois números primos aleatórios, p e q. Calcula-se $n=pq$.

Aleatoriamente escolhe-se a chave de encriptação *e,* tal que *e* e $(p - 1)(q - 1)$ sejam primos relativos.

A chave de desencriptação *d* é o inverso de *e*, módulo $(p - 1)(q - 1)$:

$$ed \equiv 1 \pmod{(p - 1)(q - 1)}.$$

Apenas existe se *e* e $(p - 1)(q - 1)$ forem primos relativos.

Se dois utilizadores pretenderem a troca de informação através do método RSA:

Se necessário o utilizador A envia a sua chave pública (n,e) para o utilizador B. O utilizador B encripta a mensagem através da chave pública de A:

$$C = P^e \bmod n,$$

enviando "C" para o utilizador A. A desencriptação é feita através da chave privada do utilizador A, "d",

$$P = C^d \bmod n,$$

onde P = mensagem original e C = mensagem encriptada.

2.4.2 Algoritmo *Diffie-Hellman*

Notas auxiliares: Raízes primitivas e logaritmos discretos

Uma raiz primitiva de um número primo p é um número a, tal que os números

$a \bmod p, a^2 \bmod p, \ldots, a^{p-1} \bmod p$

sejam a permutação de $1, \ldots, p-1$.

Seja b um número primo e b uma raiz primitiva de p. O logaritmo discreto de base b, módulo p, de um número x é o inteiro y tal que

$b^y \bmod p = x.$

O logaritmo discreto de 1, em qualquer base, é zero, já que

$b^0 \bmod p = 1,$

e o logaritmo discreto de b, na base b, é um, já que

$b^1 \bmod p = b.$

Inicialmente são conhecidos a ambos os utilizadores, dois números p e b, que correspondem respectivamente a um número primo e a uma das suas raízes primitivas.

O utilizador gera uma chave secreta (S_A) tal que $S_A < p$ e calcula a sua chave pública $P_A = b^{S_A} \bmod p$.

O utilizador B gera também uma chave secreta (S_B) em que $S_B < p$, calculando depois a sua chave pública $P_B = b^{S_B} \bmod p$.

Os dois utilizadores fazem troca das suas chaves públicas, P_A e P_B, através do canal público.

O número que o utilizador A obtém é

$$(P_B)^{S_A} \bmod p = (b^{S_B} \bmod p)^{S_A} \bmod p$$
$$= (b^{S_B})^{S_A} \bmod p$$
$$= b^{S_B S_A} \bmod p$$

e o número que o utilizador B obtém é

$$(P_A)^{S_B} \bmod p = (b^{S_A} \bmod p)^{S_B} \bmod p$$
$$= (b^{S_A})^{S_B} \bmod p$$
$$= b^{S_A S_B} \bmod p$$

Ambos os utilizadores obtêm o mesmo valor, representado por K:

$K = (P_B)^{S_A} \bmod p = (P_A)^{S_B} \bmod p$.

Para um atacante conseguir quebrar o algoritmo de *Diffie-Hellman*, tem de calcular os logaritmos discretos S_A e S_B de P_A e P_B, que será uma tarefa muito difícil.

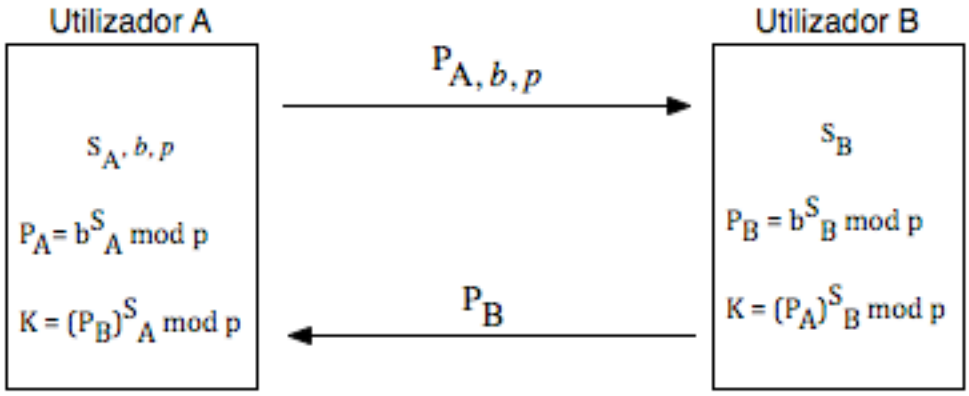

Figura 8 - Método de *Diffie-Hellman*

O algoritmo de *Diffie-Hellman* torna-se mais eficiente quanto maior for o número primo escolhido. Se o número primo for muito grande a quantidade de raízes primitivas desse número também aumenta, sendo mais difícil para um atacante descobrir o resultado.

2.5 *VoIP*

A voz sobre *Internet* (*Voice over Internet Protocol* - *VoIP*) é uma tecnologia que permite ao utilizador estabelecer chamadas telefónicas através de uma rede de dados, convertendo um sinal de voz analógico num conjunto de sinais digitais, posteriormente enviados através de uma ligação à Internet sob a forma de pacotes com endereçamento *IP*. Consiste no uso das redes de dados que utilizam o conjunto de protocolos *TCP/UDP/IP* para a transmissão de sinais de voz em tempo real na forma de pacotes. Uma das suas vantagens é que as chamadas telefónicas de voz

circulam juntamente com outro tipo de informação, evitando os custos que essas mesmas chamadas teriam se fossem enviadas isoladamente através da rede.

Para que a transmissão de voz seja possível, o *VoIP* captura a voz, digitaliza-a e de seguida transforma-a em pacotes de dados, que podem ser enviados pela rede usando os protocolos *TCP/IP* (*Transport Control Protocol/Internet Protocol*). Assim, é perfeitamente possível trabalhar com esses pacotes pela internet. Quando o destino recebe os pacotes, estes são transformados em sinais analógicos e enviados para um dispositivo no qual seja possível ouvir o som.

Para que o *VoIP* se tornasse numa tecnologia viável, foi necessário investir em *QoS* (*Quality of Service*), isto é, em qualidade de serviço. Dadas as características do sinal de voz são necessários 64kb/s para a transmissão com qualidade, correspondentes a uma frequência de amostragem de 8kHz e a utilização de uma quantificação pseudo-logarítmica de 8b/amostra. Este é o ritmo de base em comunicações telefónicas e está definido na Recomendação G711 da *ITU-T* (*International Telecommunications Union - Telecommunications standardization sector*). Actualmente esta taxa de transmissão não constitui qualquer problema com o acesso generalizado à Internet de banda larga, principalmente em empresas. Mas no início da utilização de redes digitais foi necessário desenvolver codecs apropriados para reduzir a taxa de transmissão, mantendo a qualidade do sinal de voz (inteligibilidade e fácil identificação do locutor).

Os codecs são protocolos extras que adicionam funcionalidades e maior qualidade à comunicação. Entre eles, tem-se o G.711 (já referido), o G.722, o G.723, o G.727 e o GSM6.10 (usado nas redes europeias de telemóveis) da *ITU-T*. O que os diferencia são os algoritmos usados, o atraso introduzido, a taxa de transmissão e, principalmente a qualidade da voz. Neste último aspecto, o G.711 é considerado excelente e constitui a referência dos codecs em termos de qualidade.

Para que seja possível a interligação das redes telefónicas convencionais com o *VoIP* usa-se um equipamento denominado de *Gateway*. Este é responsável por fazer a conversão do sinal analógico em digital e vice-versa, fazendo também a conversão da sinalização auxiliar das chamadas telefónicas.

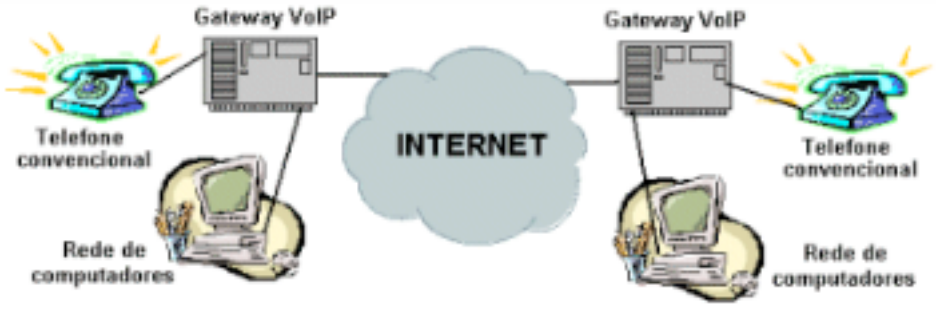

Figura 9 - Modelo de arquitectura *VoIP*

Capítulo 3 Proposta do sistema

3.1 Proposta do sistema de criptografia assimétrica

Por princípio é escolhido um número primo *p* e uma das suas raízes primitivas *b*. Estes valores são conhecidos de ambos os utilizadores e assim inicia-se o método de *Diffie-Hellman*.

O utilizador A introduz a sua chave secreta (S_A) e através dela gera a sua chave pública (P_A). Em paralelo o utilizador B introduz também a sua chave secreta (S_B) e gera a sua chave pública (P_B). De seguida o utilizador A envia a sua chave pública para o utilizador B e fica á espera de receber a chave pública do utilizador B. Quando o utilizador A recebe P_B faz o cálculo de K, assim como o utilizador B. O valor de K será igual nos dois utilizadores.

A seguinte figura demonstra o processo explicado.

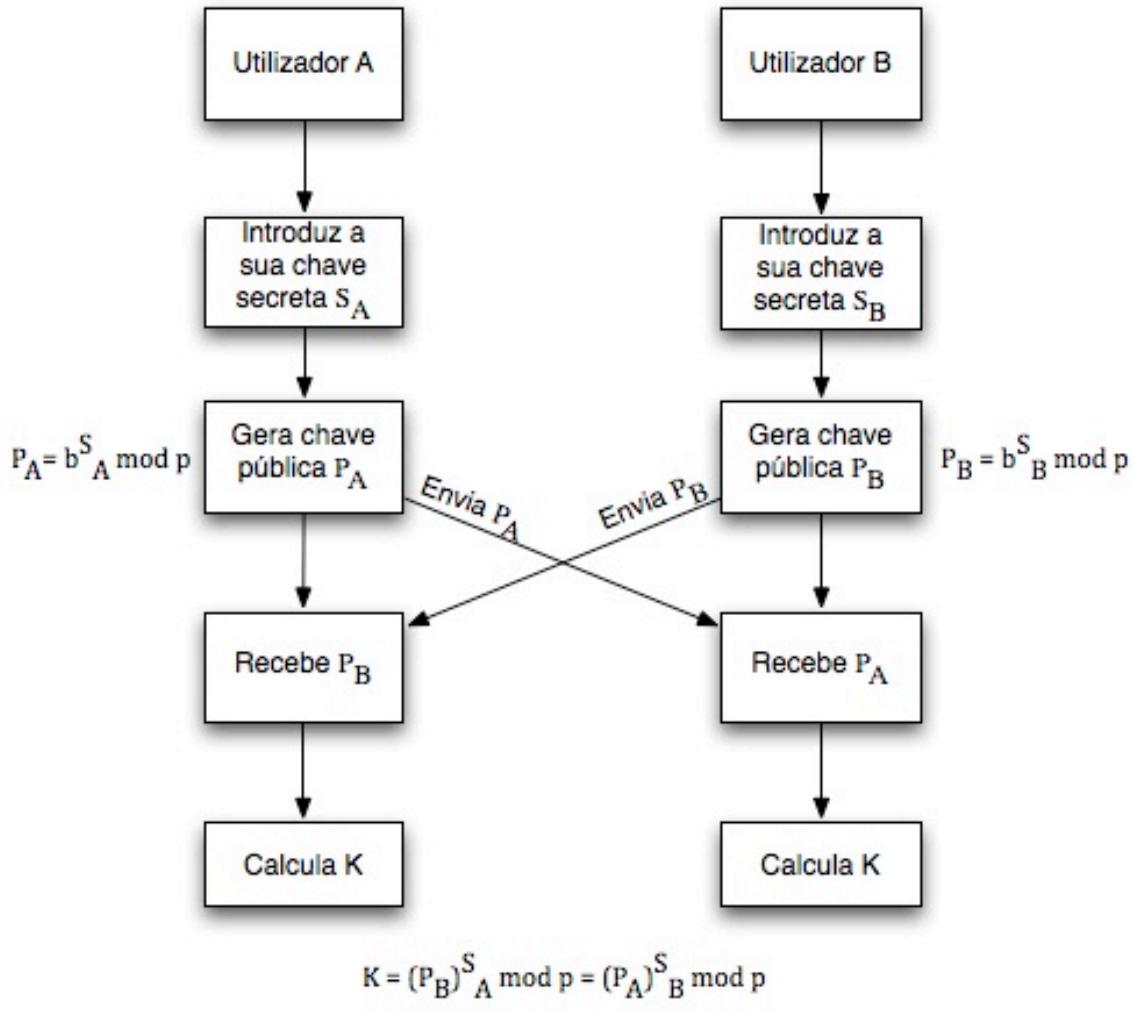

Figura 10 - Proposta do sistema de *Diffie-Hellman*

3.2 Proposta do sistema de *scrambling*

A figura que se segue explica a segunda parte do sistema.

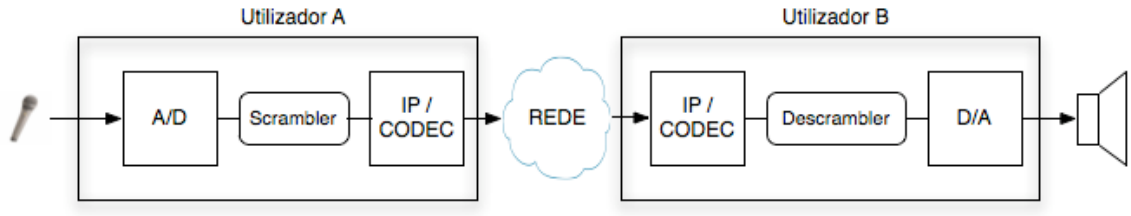

Figura 11 - Proposta do sistema de *scrambling* digital e codec

Os utilizadores A e B através do método de *Diffie-Hellman* geram uma chave privada que apenas os dois conhecem. De seguida, aplica-se a técnica de *scrambling* ao sinal digital com base na chave gerada K anteriormente. O segmento de sinal de voz alterado é enviado pelo canal de voz público (*VoIP*), onde vai ser recebido pelo utilizador B. Este utilizador através da chave gerada pelo método de *Diffie-Hellman* descodifica o segmento de sinal de voz e assim consegue reproduzi-lo.

Capítulo 4 Implementação

4.1 Código de chave pública/privada

Nesta implementação o método usado foi o de *Diffie-Hellman*, sendo que este é o que se adapta melhor para a troca de chaves públicas e gerar uma chave privada que apenas os dois utilizadores conhecem.

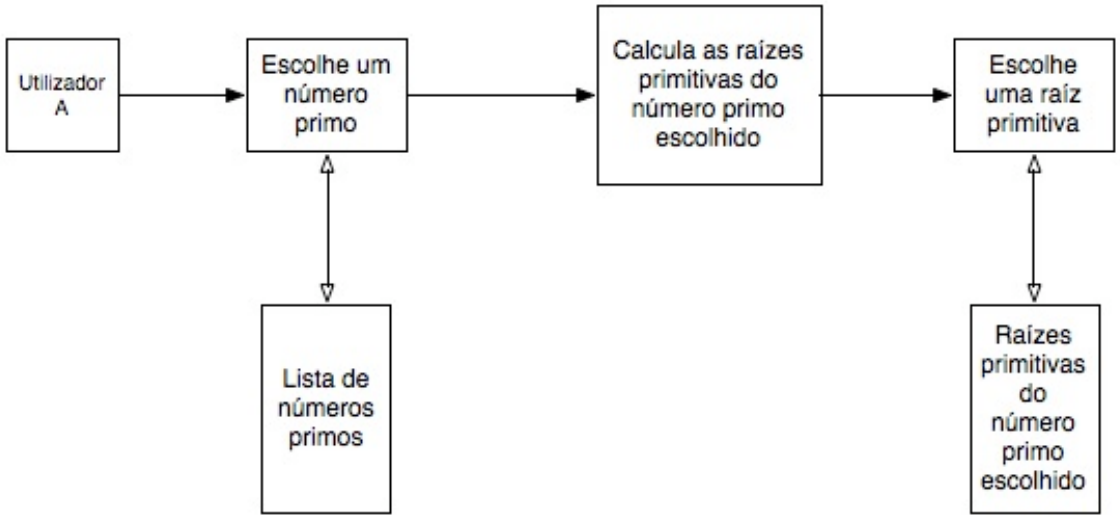

Figura 12 - Escolha do número primo e raíz primitiva

A figura anterior mostra que é o utilizador A o responsável pela determinação do número primo e da sua raiz primitiva.

Na implementação em Matlab recorreu-se ao uso de uma função já desenvolvida, para o cálculo das raízes primitivas de um número primo. Esta função apenas calculava uma raiz primitiva de um número primo, sendo alterada para efectuar o cálculo de todas as raízes primitivas de um número primo.

Outro problema que surgiu foi o uso da função "*mod()*" no Matlab, pois se o número primo escolhido for muito elevado a sua raiz primitiva também será, resultando em operações muito grandes quando se calculam módulos de potências. A função que calcula o módulo no Matlab é bastante limitada, tornando-se impossível o cálculo de módulos nestas situações. Recorreu-se a uma função já desenvolvida com o nome de "*bigmod()*".

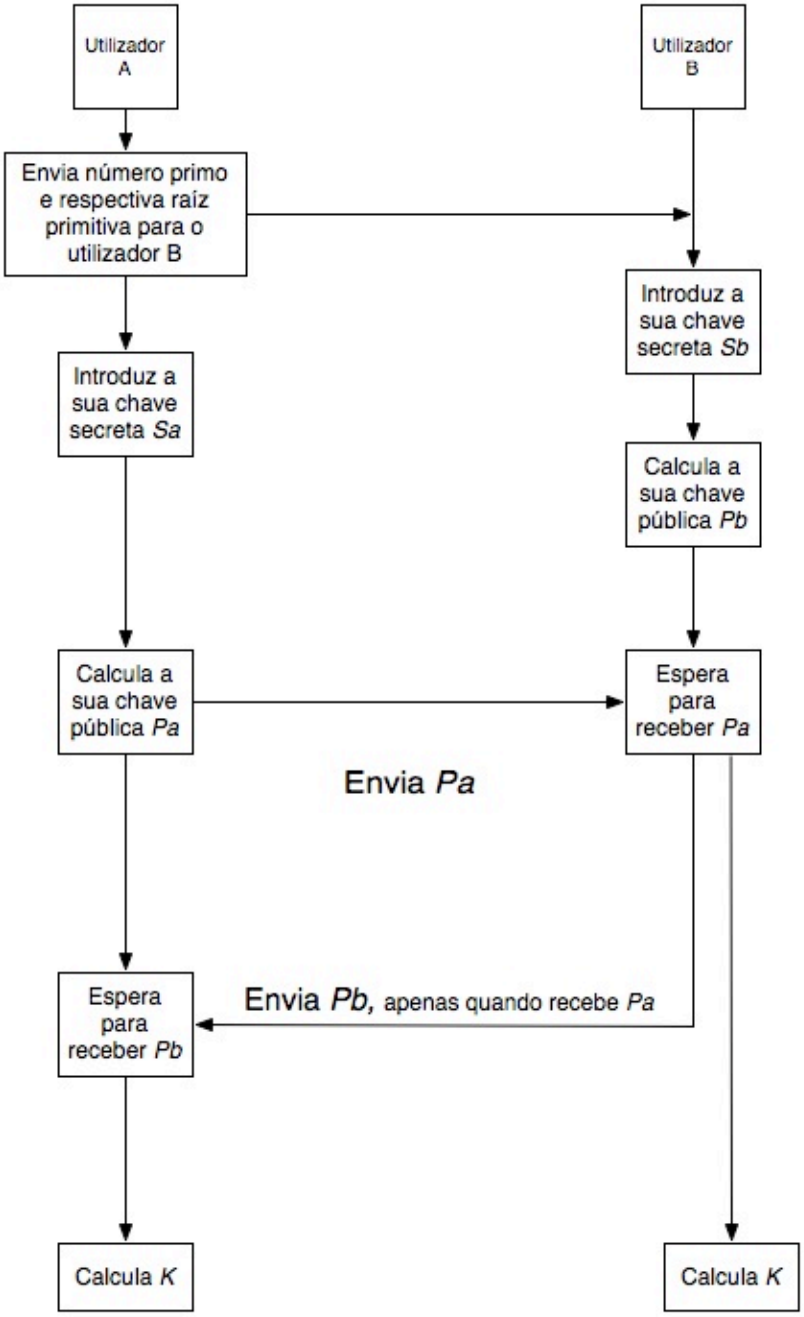

Figura 13 - Implementação do método de *Diffie-Hellman*

No esquema anterior são explicados todos os passos necessários para a implementação do programa.

Um aspecto relevante a referir é que a troca de chaves pública entre os dois utilizadores são feitas através do protocolo *TCP* (*Transmission Control Protocol*). Este é um protocolo do nível de camada de transporte. É uma conexão ponto a ponto, estabelecida entre dois pontos e full-duplex, em que é possível a transferência

simultânea em ambas as direcções. Este protocolo garante que no final da conexão todos os pacotes sejam bem recebidos.

Para se fazer a troca das chaves públicas entre ambos os utilizadores é necessário conhecer ambos os *IP*'s dos utilizadores. O utilizador A precisa de saber o *IP* do utilizador B quando envia a sua chave pública e vice versa.

4.2 *Scramble* e *Descramble* das Frequências

A proposta de *scrambler* que se faz neste trabalho é no domínio das frequências. Propõe-se que, para cada segmento, se reordenem as frequências de uma forma pseudo-aleatória dependendo de uma chave conhecida pelos dois utilizadores. Calcula-se a DFT do segmento e em primeira aproximação faz-se a reordenação pseudo-aleatória das frequências (notar que a simetria da Transformada de Fourier na frequência deve ser mantida) e de seguida calcula-se a Transformada inversa de Fourier, obtendo-se o sinal *scrambled*. Isto origina o problema espectral já referido: o sinal *scrambled* pode ter um espectro muito diferente do de um sinal de voz.

A nossa proposta é fazer-se a divisão em altas e baixas frequências e de seguida fazer o *scramble* das baixas e das altas independentemente. Como o espectro do sinal de voz é em geral muito mais rico nas baixas frequências, garante-se desta forma um espectro do sinal *scrambled* ainda com essa característica (A Figura 14 confirma esta afirmação).

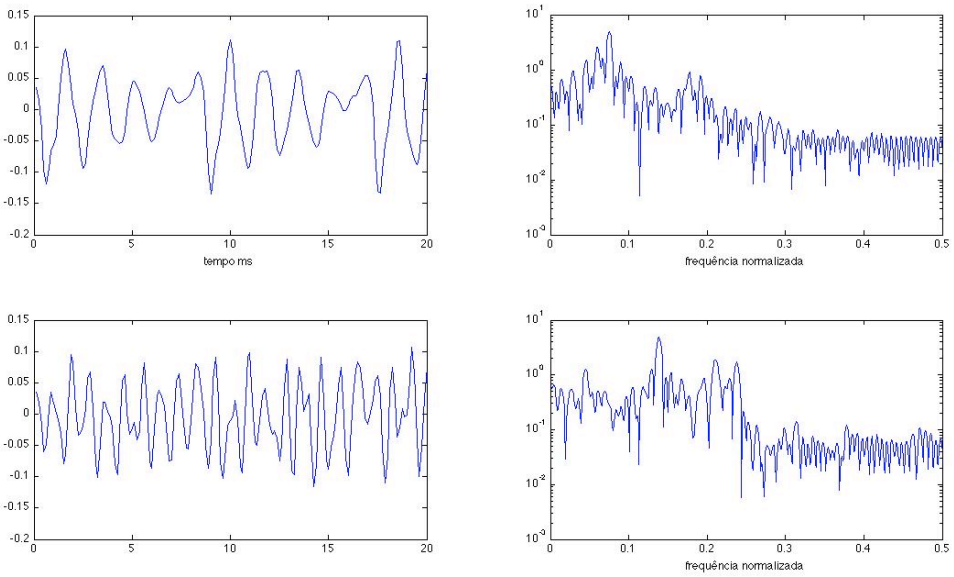

Figura 14 - *Scramble* do sinal

Para implementar o algoritmo de *scramble* são necessários os seguintes passos:

1. Segmentar o sinal de entrada com N amostras;
2. Determinar a DFT do segmento;
3. Determinar, baseado no K conhecido dos dois utilizadores qual a permutação das componentes na frequência se vai utilizar;
4. Efectuar separadamente o reordenamento das baixas frequências ( ]*0, fs/4*[ ) e das altas frequências ( ]*fs/4, fs/2*[ );
5. Determinar as componentes nos intervalos ( ]*fs/2, 3fs/4*[e]*3fs/4, fs*[ ) usando as propriedades de DFT para sinais reais - simetria conjugada em torno de fs/2;
6. A componente contínua mantém-se;
7. Determinar a DFT inversa, obtendo um sinal real.

Para efectuar o contrário, *descrambler,* usa-se exactamente o mesmo procedimento, apenas se substitui a permutação de amostras pela permutação inversa.

A implementação em Matlab necessita então dos seguintes passos:

1. O utilizador A calcula a variável "per" de permutação, com base no valor de K, determinado anteriormente e usando a função *sort()*. A variável *per* tem tamanho de ((N/4)-1) , em que N é o número de amostras por cada segmento e contém a ordem pseudo-aleatória das amostras. O valor de K resulta do algoritmo de *Diffie-Hellman*, sendo conhecido por ambos os utilizadores, e vai ser usado para inicializar a função rand():
   >>*rand('twister',k)*
   permitindo assim que o utilizador B, que conhece K, também saiba a ordem de permutação;
2. O sinal de entrada "teste.wav" é segmentado com N amostras;
3. Determina-se a DFT do segmento através da função *fft()*;
4. Efectua-se separadamente o reordenamento das baixas frequências e altas frequências;
5. Através das funções *conj()* e *fliplr()* determinam-se as componentes do sinal nos intervalos ( ]fs/2, 3fs/4[e]3fs/4, fs[ ), através da simetria conjugada;
6. Calcula-se a transformada inversa de Fourier através da função *ifft()* e o sinal *scrambled* será apenas a parte real do resultado anterior.

A função chama-se "*scramblingnew*" e tem uma saída que é "*xscr*" que corresponde ao sinal *scrambled* e três entradas: "*N*" que é o número de amostras que contém cada segmento, "*x*" que é o sinal de entrada e "*per*" a variável que contém o ordenamento das amostras.
A função de *descramble* é igual, mudando uma das entradas, que em vez de ser "*per*" é "iper". A variável "*iper*", calculada através da função *find()*, contém o ordenamento contrário das amostras em relação à variável "*per*".

4.3 Simulação da ligação em tempo real

Uma vez que não se dispõe de microfones/altifalantes disponíveis para ligar a um PC e respectiva interface com o Matlab, foi decidido que apenas se usaria um processo de simulação. Assim o sinal de entrada está num ficheiro no computador A e poderá ser lido pelo programa Matlab em segmentos de N amostras, procedendo-se em seguida ao seu *scrambling* conforme o algoritmo anterior.

Através de um *socket* (define-se como a combinação de um endereço *IP*, um protocolo e o número da porta do protocolo; permite a transferência de pacotes de um computador para outro via *TCP*) o programa envia o resultado em segmentos *scrambled* para um outro computador B, identificado por um endereço *IP*. No computador B um programa fará o *descramble* dos segmentos e colocará o resultado num ficheiro. Este ficheiro será audível usando o comando "*sound*".

Um dos problemas que surge é a necessidade de se usar caracteres na transmissão por *TCP*. Isso obriga a uma conversão do sinal interpretado como números inteiros para caracteres. Na recepção será feita a operação inversa.

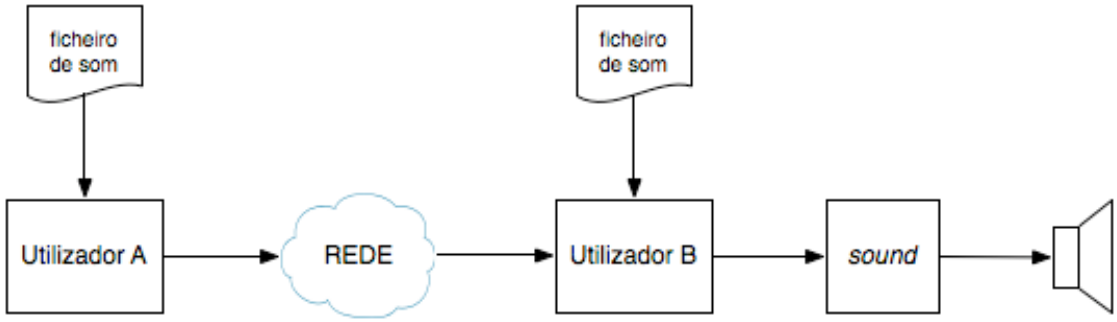

Figura 15 - Simulação da ligação

4.4 Funcionamento com outros codecs

Para o uso de outros codecs neste sistema, recorreu-se a um programa externo denominado *Sox*. Este programa tem a finalidade de fazer conversões do ficheiro de voz (que por defeito o formato é wav) para outros formatos, tais como "gsm", etc.., para verificar como se comporta o sistema perante outros codecs.

Aplica-se o *scramble* a um ficheiro de voz, com o formato "wav" , de entrada e guarda-se o ficheiro alterado, isto é, *scrambled*. Através do programa Sox converte-se o ficheiro para um formato arbitrário e de novo converte-se para "wav". De seguida aplica-se o *descrambler* a este ficheiro e assim obtém-se o sinal reconstruído.

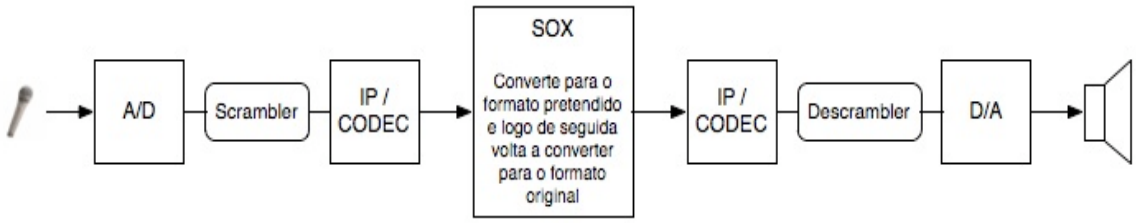

Figura 16 - *Scrambling* com codec através do *Sox*

O uso do *Sox* é bastante simples e como tal segue-se o exemplo:

>>sox teste.wav testegsm.gsm

>>sox testegsm.gsm testegsm.wav

O primeiro comando converte o formato wav para o formato gsm e o segundo comando o inverso. Assim, faz-se o teste para diferentes codecs.

Neste caso não se faz o uso da rede, uma vez que se pretende apenas verificar o desempenho do sistema com os codecs sendo desnecessário o uso desta, e assim o esquema apresentado em cima foi efectuado em apenas um computador.

4.5 Codecs

Os codecs do tipo forma de onda procuram acompanhar a forma de onda do sinal. Os mais conhecidos são o PCM – modulação por codificação de impulsos (*Pulse Code Modulation*) e ADPCM – modulação por codificação diferencial adaptativa de impulsos (*Adaptive Pulse Code Modulation*).

No PCM as amostras são representadas directamente por um código com um certo número de bits. O PCM mais importante é de facto o primeiro sistema standardizado de codificação e definido na recomendação G711 de *ITU*, e como já referido, é uma codificação que resulta de uma quantificação pseudo-logarítmica de 8bit/amostra de um sinal digitalizado de 14bit/amostra. A referida norma define as tabelas de conversão que são usadas (nota que, como a quantificação usada nos EUA e na Europa não é a mesma, também são definidas em tabelas de conversão).

O ADPCM codifica não o sinal mas a primeira diferença definida por:

$$y(n) = x(n) - x(n-1), \text{ para } n = 1,2,3,\ldots \text{ e } x(0) = 0.$$

A vantagem de se trabalhar com a diferença reside no facto de ser demonstrável que são precisos menos bits por amostra para codificar y(n) que x(n). Sobre isto ainda se aplica um processo adaptativo de predição do sinal que aumenta a eficácia da codificação.

Assim se para o PCM se usam 64Kb/s para o APCM são possíveis taxas de transmissão a partir dos 24Kb/s, mantendo-se pouco alterada a qualidade do sinal.

Os vocoders são codecs baseados num modelo de produção de voz. Esse modelo é o modelo de predição linear em que cada amostra é considerada como sendo aproximadamente uma combinação linear das amostras anteriores

$$x(n) \approx \sum_{k=1}^{p} a_k x(n-k)$$

Para cada segmento são calculados os a_K – coeficientes de predição linear (LPC – *Linear Prediction Coeficient*) que são as quantidades transmitidas por segmento.

Na recepção, usando os a_K recebidos reconstrói-se um sinal que se aproxima do sinal original.

Para além dos a_K pode transmitir-se mais informação e é o tipo de informação extra que se transmite que caracteriza o *vocoder*. Iremos usar dois *vocoders* : o LPC10 e o GSM6.10.

O LPC10 é uma norma americana militar que permite uma taxa de transmissão de 2.4Kb/s com baixa qualidade.

O GSM6.10 é a norma utilizada no sistema europeu de telemóveis e permite uma taxa de transmissão de 13Kb/s com alta qualidade.

No entanto em qualquer dos *vocoder* se o sinal a codificar se afastar muito do seu sinal de voz, a qualidade degrada-se fortemente.

Capítulo 5 Resultados

Como o objectivo é reproduzir o sinal original de uma forma inteligível, a avaliação da qualidade da codificação/descodificação pode ser avaliada de duas maneiras: objectivamente medindo o grau de semelhança de *s(n)* e *s'(n)* ou subjectivamente avaliando a inteligibilidade de *s'(n)*.

As medidas objectivas mais usadas são a relação sinal ruído SNR e a relação sinal ruído segmental SSNR

$$\mathrm{SNR}=10\log_{10}\frac{\sum_n s^2(n)}{\sum_n (s(n)-s'(n))^2},$$

e supondo que o sinal se encontra dividido em M segmentos com N amostras

$$\mathrm{SSNR}=\frac{1}{M}\sum_{i=0}^{M-1}10\log_{10}\frac{\sum_{n=0}^{N-1} s^2(iM+n)}{\sum_{n=0}^{N-1}(s(iM+n)-s'(iM+n))^2}$$

A avaliação objectiva apresenta problemas pois dois sinais que difiram de um valor constante podem apresentar um valor significativo de SNR sendo contudo perfeitamente iguais do ponto de vista perceptual. Por essa razão as medidas subjectivas que olham para a percepção humana do sinal são em geral mais interessantes de usar.

O teste subjectivo mais utilizado é o MOS (*Mean Opinion Score*) que consiste em avaliar a voz codificada/descodificada por ouvintes humanos que classificam o que ouvem com uma pontuação 1 (muito mau, ininteligível) a 5 (excelente, não distinguível de voz não sujeita ao processo de codificação). O resultado é em geral a média de *n* ouvintes e é muitas vezes acompanhado de um intervalo de confiança.

Feita esta introdução, apresenta-se de seguida o sinal de entrada usado para os testes que se seguem. Tem o nome de "teste.wav", foi gravado no formato wav e tem uma frequência de amostragem de 8KHz e tem uma duração de aproximadamente 11,9 segundos.

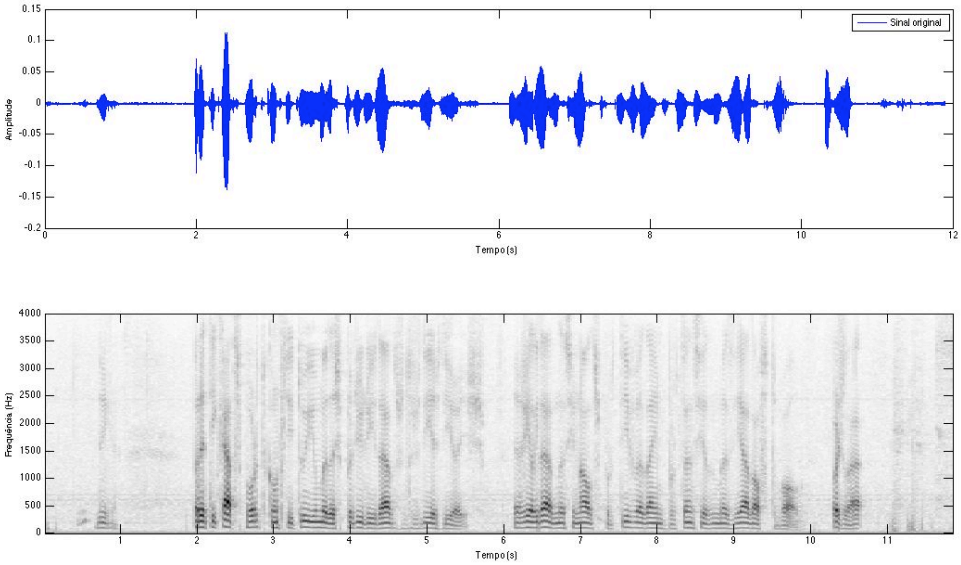

Figura 17 - Sinal original

Os testes que vão ser feitos são os seguintes:

- Usar o *scrambler/descrambler* propostos;
- Usar apenas os Codecs;
- Usar os dois anteriores em conjunto;
- Introduzir ruído ao sinal.

Os resultados apresentados pela avaliação MOS são a média da classificação de quatro ouvintes.

5.1 *Scramble/Descramble*

O *s(n)* determina-se por cada segmento de 160 amostras. O *s'(n)* é determinado entre a diferença do sinal original e o sinal final. Calcula-se o *s'(n)* para cada segmento e de seguida faz-se o cálculo do SSNR.

SSNR = 118.3 dB

MOS = 4.5

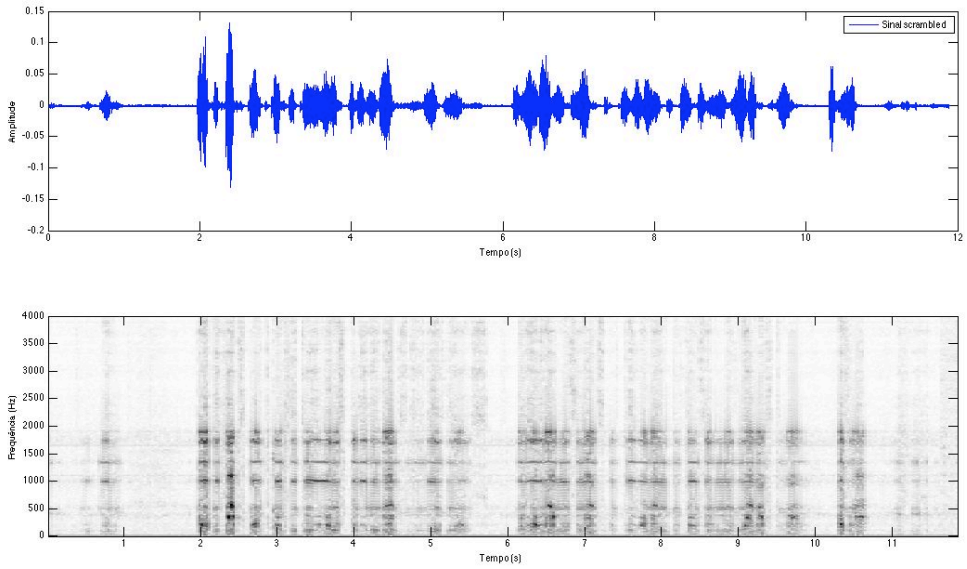

Figura 18 - Sinal *scrambled*

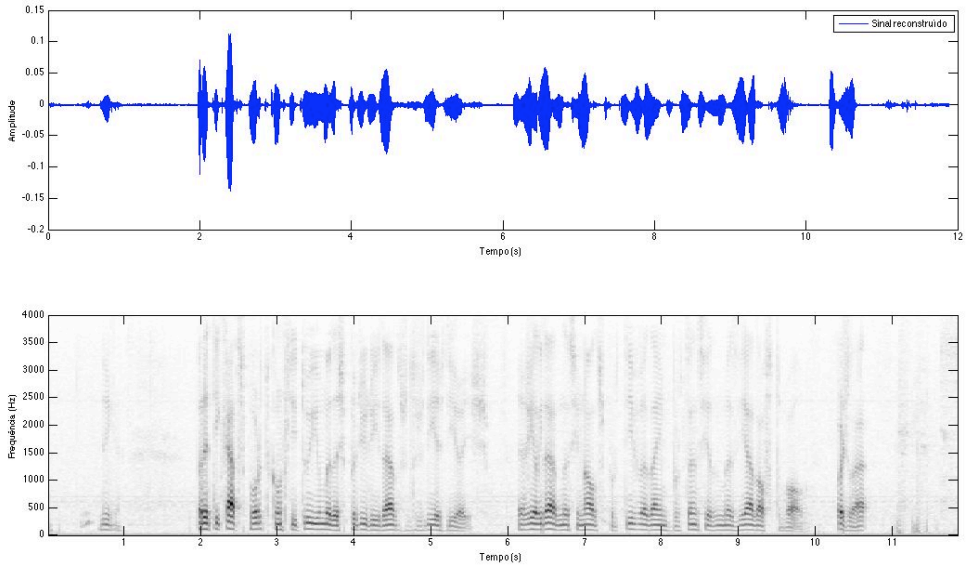

Figura 19 - Sinal *descrambled*

O SSNR tão elevado significa apenas que há ruído aritmético introduzido.

5.2 Codecs

Primeiro faz-se uma análise ao ruído que apenas é produzido pelo codec e procede-se da seguinte maneira:

- Faz-se uma análise do sinal original por segmentos de 160 amostras cada e de seguida é calculada a sua energia;
- De seguida analisa-se o sinal com o efeito do codec, por segmentos de 160 amostras;
- Determina-se o erro por diferença entre o sinal de entrada e o sinal com efeito do codec;
- Calcula-se a energia do erro;
- Calcula-se o SSNR;

5.2.1 WAVPCM

SSNR = Infinito

MOS = 5.0

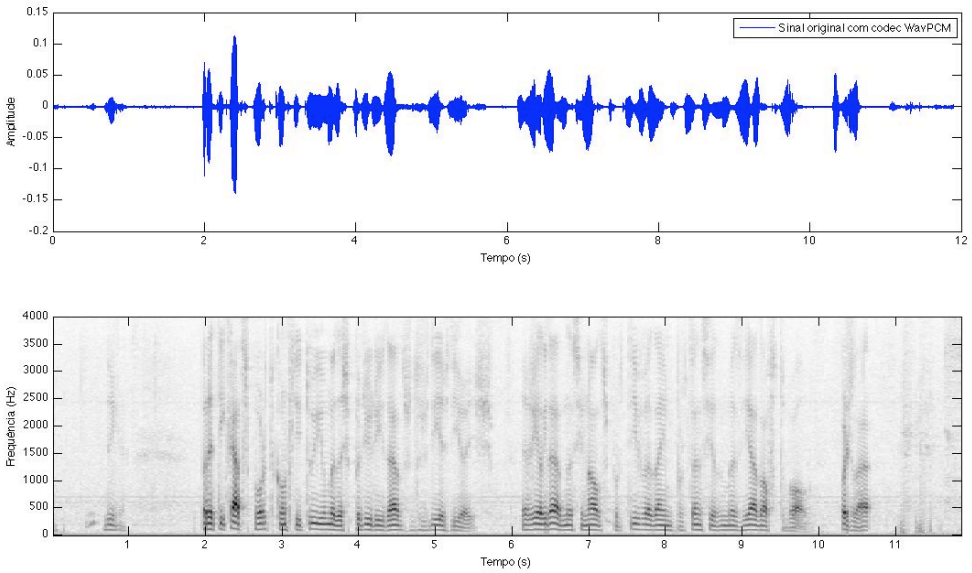

Figura 20 - Sinal com efeito do codec *WAVPCM*

5.2.2 VOX

SSNR = 33.2 dB

MOS = 3.7

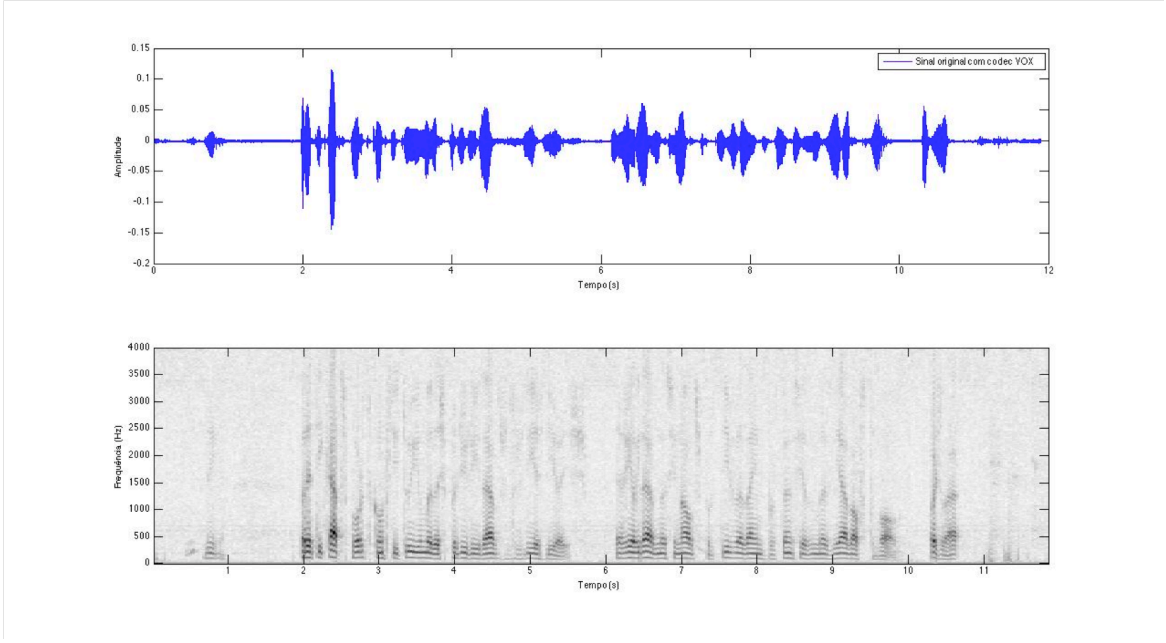

Figura 21 - Sinal com efeito do codec *VOX*

5.2.3 GSM

SSNR = 15.8 dB

MOS = 4.5

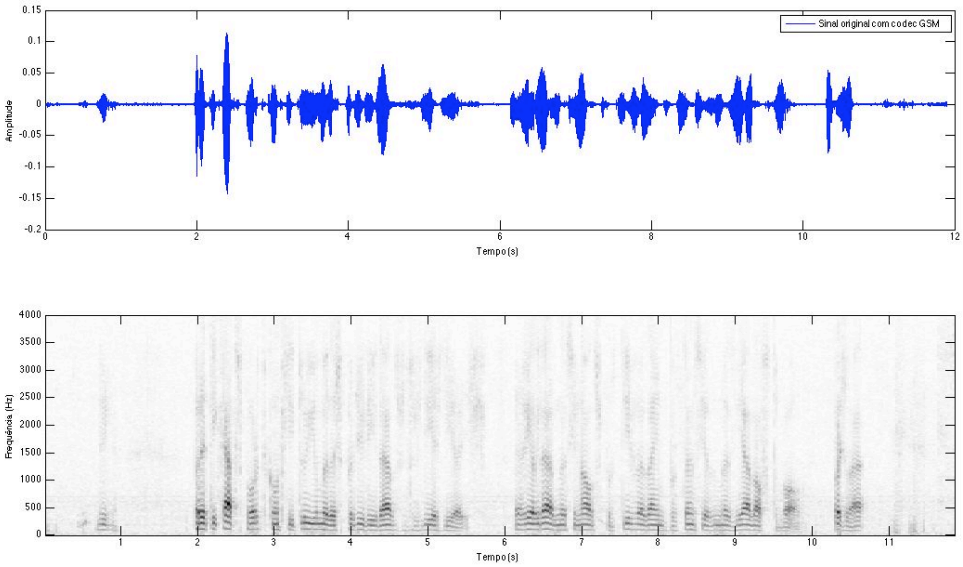

Figura 22 - Sinal com efeito do codec *GSM*

5.2.4 LPC10

Para este codec, cada segmento contém 180 amostras.

SSNR = - 7.9 dB

MOS = 3.1

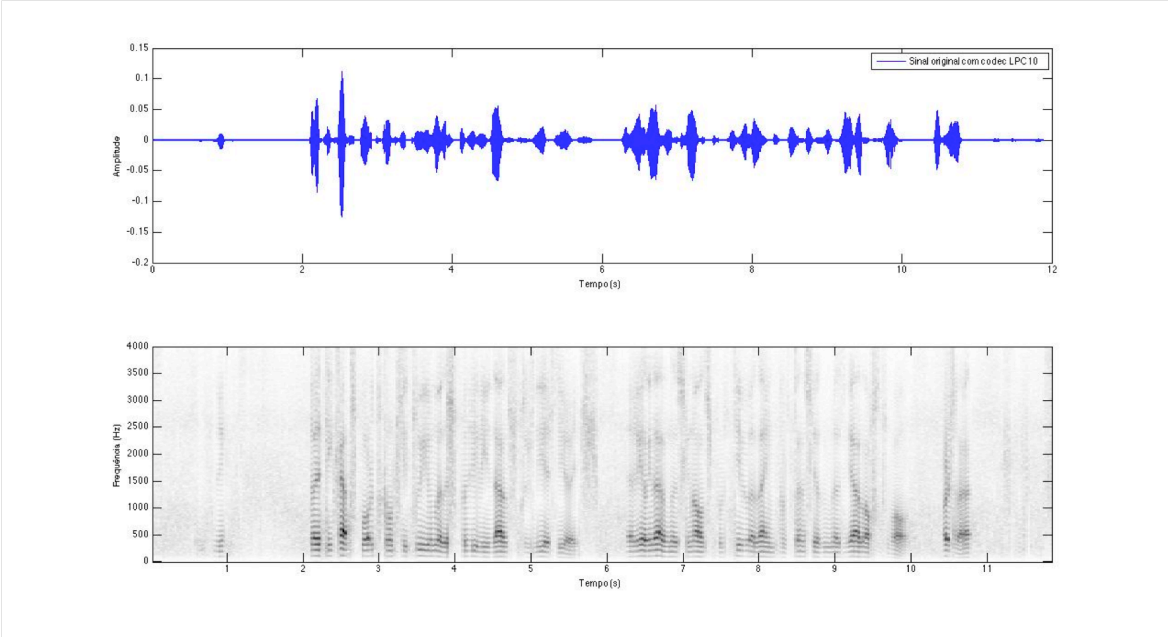

Figura 23 - Sinal com efeito do codec *LPC10*

5.3 *Scramble/Descramble* e codecs

Agora faz-se uma análise com os codecs e com o *scrambler*:

O método utilizado é semelhante ao explicado anteriormente, sendo que, em vez de se aplicar o codec ao sinal original, aplica-se o codec ao sinal *scrambled*.

5.3.1 WAVPCM

SSNR = 118.3 dB

MOS = 4.7

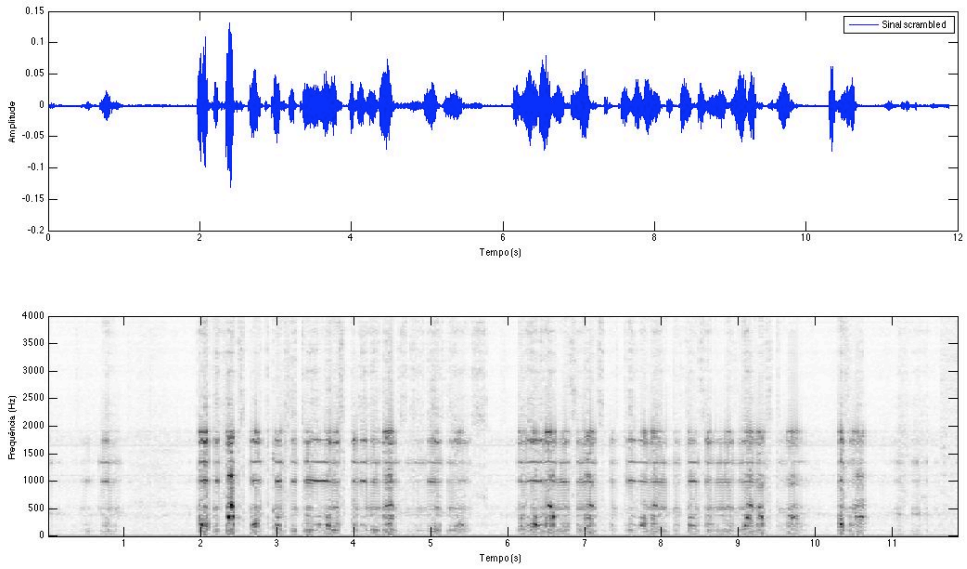

Figura 24 - Sinal *scrambled* com efeito do codec WAVPCM

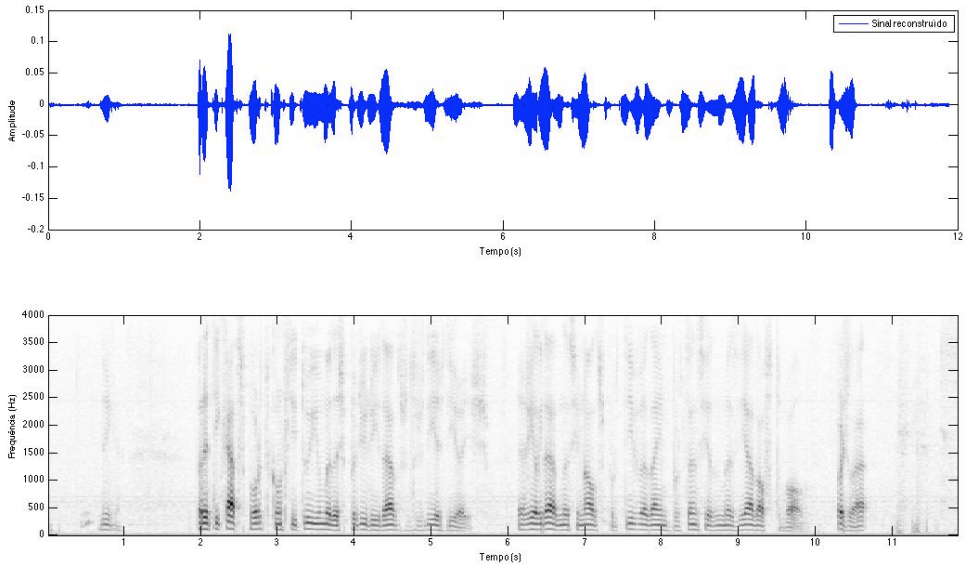

Figura 25 - Sinal *descrambled* com efeito do codec WAVPCM

5.3.2 VOX

SSNR = 29.6 dB

MOS = 3.1

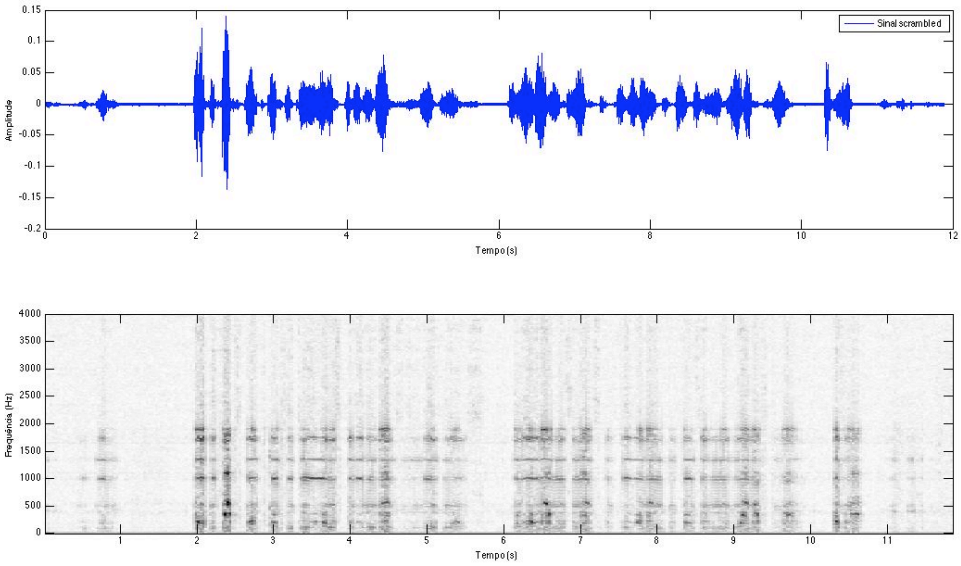

Figura 26 - Sinal *scrambled* com efeito do codec VOX

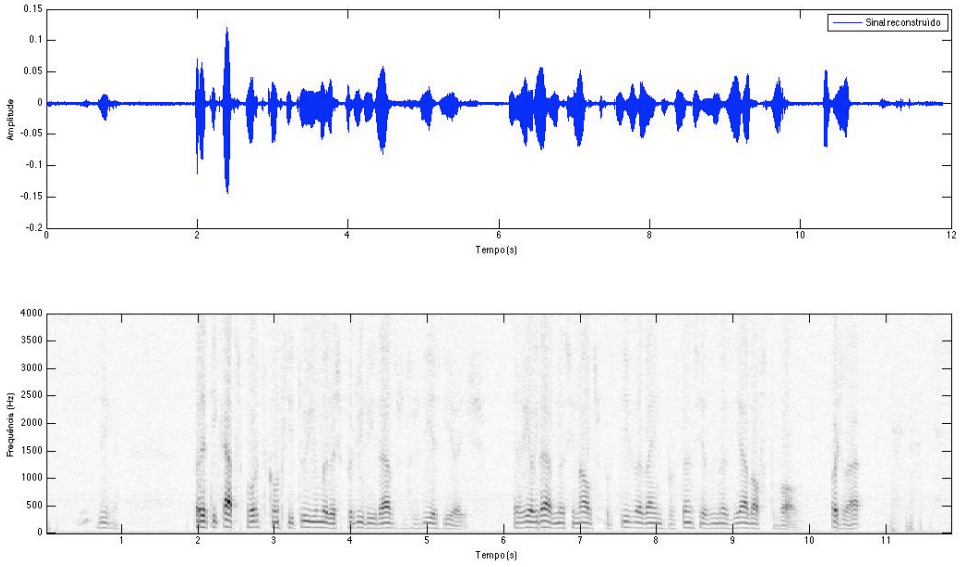

Figura 27 - Sinal *descrambled* com efeito do codec VOX

5.3.3 GSM

SSNR = 8.7 dB

MOS = 2.0

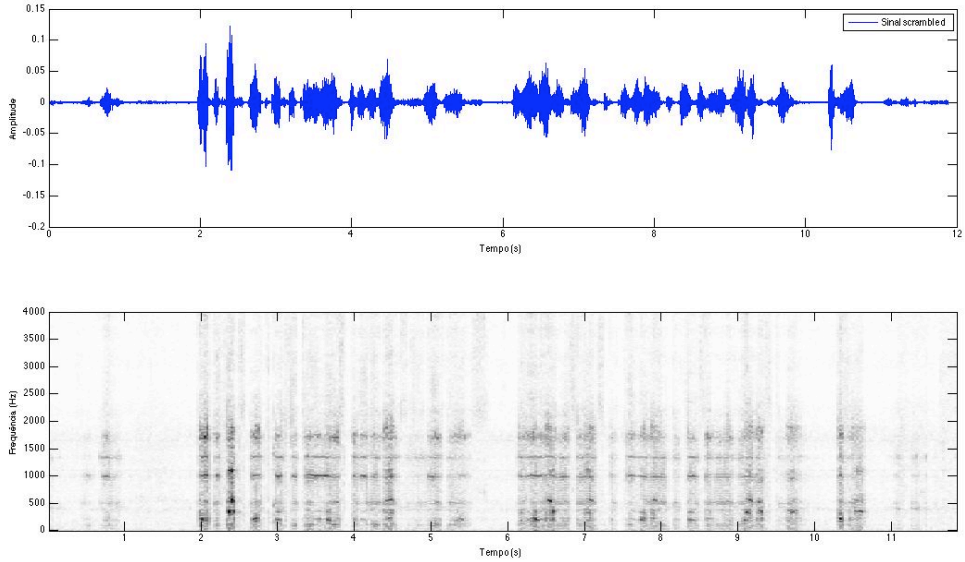

Figura 28 - Sinal *scrambled* com efeito do codec GSM

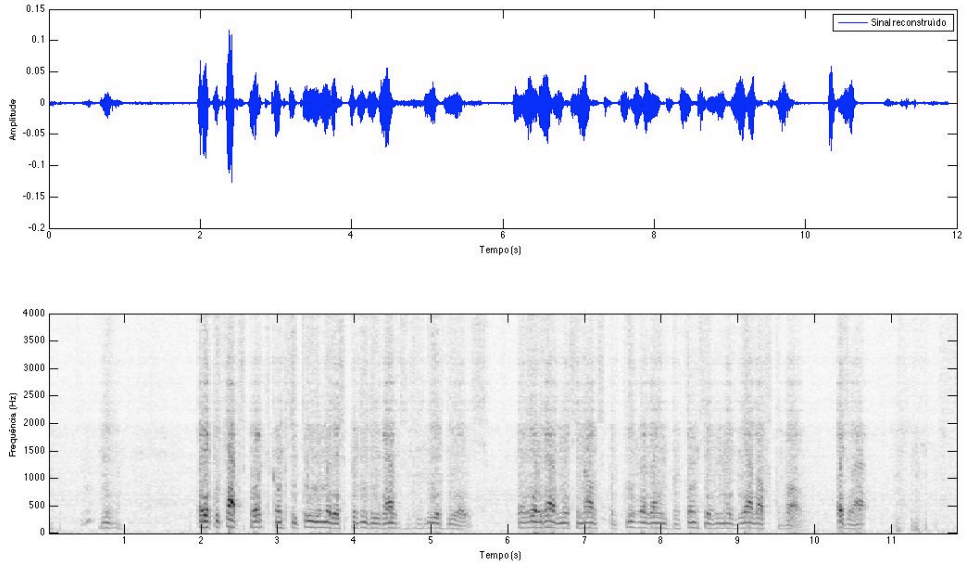

Figura 29 - Sinal *descrambled* com efeito do codec GSM

5.3.4 LPC10

Mais uma vez cada segmento contém 180 amostras.

SSNR = -10.8 dB

MOS = 1.0

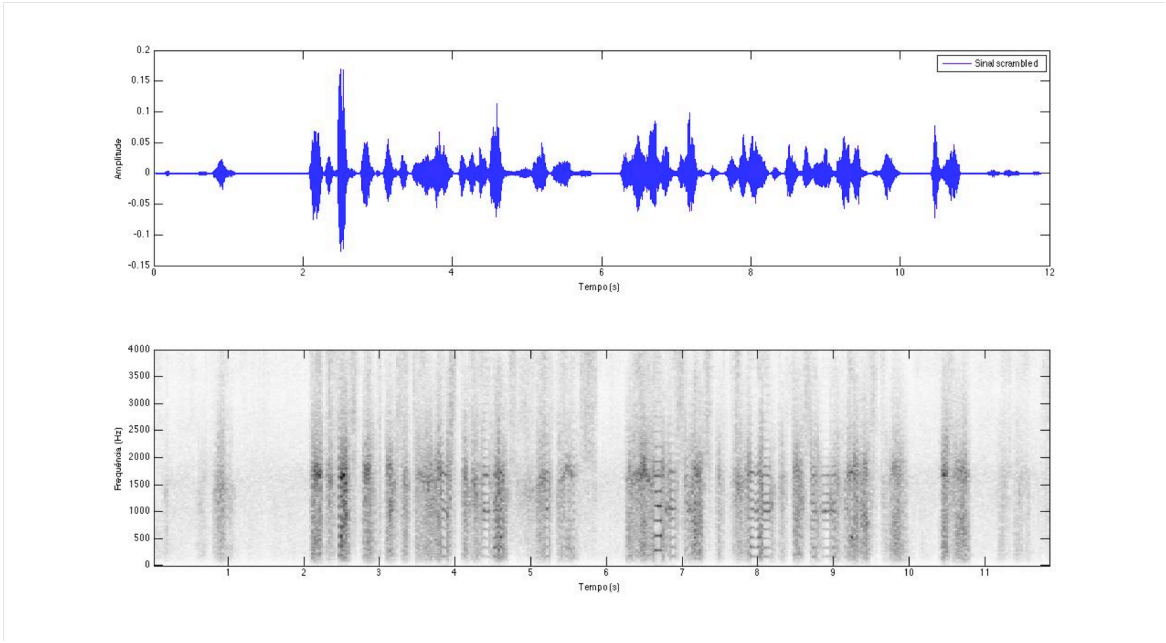

Figura 30 - Sinal *scrambled* com efeito do codec LPC10

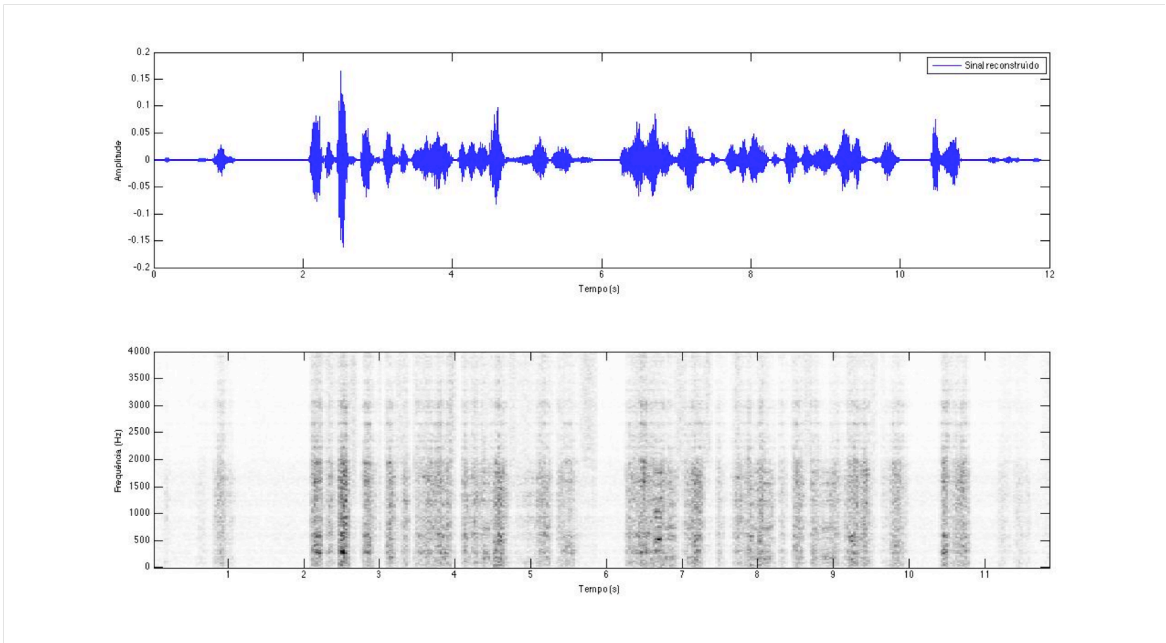

Figura 31 - Sinal *descrambled* com efeito do codec LPC10

5.4 Ruído

Por fim faz-se o teste do ruído, em que é feito o *scramble* ao sinal original por segmentos de 160 amostras e introduz-se 10 dB de ruído ao sinal *scrambled* :

SSNR = 1.6 dB

MOS = 2.7

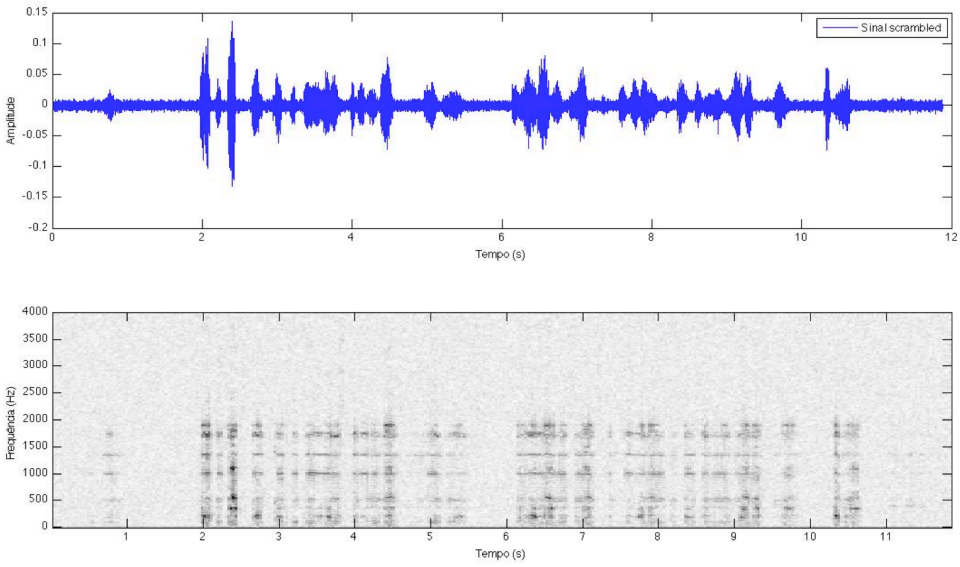

Figura 32 - Sinal *scrambled* com ruído

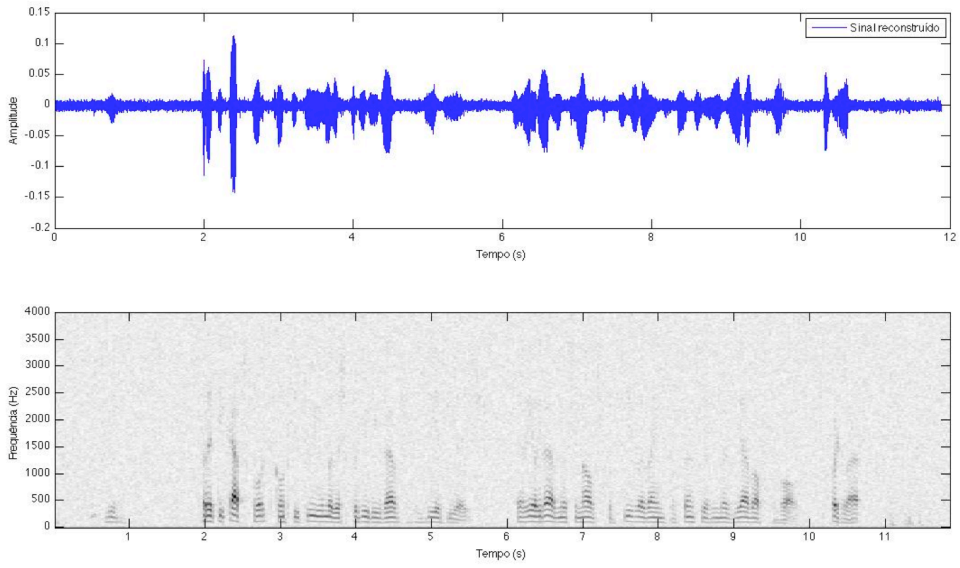

Figura 33 - Sinal *descrambled* com ruído

Testes / Avaliação	*Scramble/ Descramble*	WavPCM	VOX	GSM	LPC10	*Scramble/ Descramble* + WavPcm	*Scramble/ Descramble* + VOX	*Scramble/ Descramble* + GSM	*Scramble/ Descramble* + LPC10	*Scramble/ Descramble* + Ruído
MOS	4.5	5.0	3.7	4.5	3.1	4.7	3.1	2.0	1.0	2.75
SSNR (dB)	118.3	Infinito	33.2	15.8	-7.9	118.3	29.6	8.7	-10.8	1.6

Tabela 1 - Tabela dos Resultados

Capítulo 6 Conclusões

O primeiro teste é feito apenas para o *scramble/descramble*:

1. Pode concluir-se a não introdução de qualquer erro significativo. O resultado de 118.3 dB significa apenas que há um erro resultante das operações aritméticas envolvidas.

As conclusões que se seguem são apenas para o efeito que os codecs causam em relação ao sinal de entrada:

2. O codec WavPCM não afecta o sinal e portanto o resultado infinito é expectável.

3. O efeito do codec VOX (adpcm) apresenta pouco ruído no sinal de entrada e segundo a avaliação MOS, conclui-se que o sinal se percebe perfeitamente bem.

4. O codec GSM como já se referiu anteriormente é do tipo vocoder e degrada bastante o sinal segundo a avaliação objectiva. No entanto, o resultado apresentado pela avaliação MOS mostra que o sinal é bastante compreensível.

5. O codec LPC10 do tipo vocoder, apresenta os piores resultados. O SSNR apresenta muito ruído, não deixando de ser compreensível aquando da análise do MOS.

As conclusões que se seguem são relativas ao *scramble/descramble* com os variados codecs:

6. O resultado apresentado anteriormente apenas com *scramble/descramble* foi de 118,3 dB que é exactamente o mesmo quando se utiliza o codec WavPCM. O resultado do MOS é bastante bom, o sinal de saída não sofre alteração significativa quando comparado com o original.

7. O codec VOX do tipo forma de onda (adpcm) apresenta resultados satisfatórios. A partir do SSNR conclui-se que este codec introduz ruído, mas segundo a avaliação MOS não degrada a qualidade do sinal.

8. O codec GSM, como era de esperar, degrada um pouco o sinal e introduz ruído bastante significativo. Quando se ouve o resultado final é notória a quantidade de ruído presente, mas o sinal não deixa de ser compreensível.

9. O codec LPC10 apresenta os piores resultados, quando comparado com todos os codecs anteriores. No que diz respeito ao SSNR e MOS os resultados são péssimos, introduzindo bastante erro. O sinal final é incompreensível.

Com a introdução de ruído:

10. Com a introdução de ruído significativo era de esperar que a saída apresente também ruído. Este ruído não degrada a qualidade de voz, isto é, o sinal de saída é compreensível, o que se verifica pela avaliação MOS.

Na avaliação MOS, os ouvintes comentaram que, o valor baixo atribuído é devido ao ruído, ou seja, o sinal não é degradado, percebendo-se perfeitamente a conversa. Isto sugere que um outro tipo de teste subjectivo deveria ter sido usado.

Estes resultados mostram que a introdução de vocoders deve ser feita com cuidado: o GSM suporta este tipo de *scrambling* mas o LPC10 não é compatível. Já para os codecs do tipo forma de onda não se coloca qualquer limitação.

Contudo, conclui-se que o algoritmo de *scramble/descramble* proposto não introduz ruído significativo no sistema e comporta-se de uma forma robusta perante a introdução de ruído no canal e quando usado com vários tipos de codec.

Bibliografia

Blanchet, Gérard, e Maurice Charbit. *Digital SIgnal and Image Processing using Matlab.* ISTE Publishing Company , 2006.

Hahn, Brian, e Daniel Valentine. *Essential Matlab For Engineers and Scientists.* Elsevier Ltd., 2007.

Kay, Steven. *Intuitive Probability And Random Processes Using Matlab.* New York: Springer , 2006.

Kelly, Timothy. *VoIP For Dummies.* Canada: Wiley Publishing, Inc., Indianapolis, Indiana, 2005.

Marchand, Patrick, e O. Thomas Holland. *Graphics and GUIs with MATLAB.* Chapman & Hall/CRC, 2003.

Mitra, Sanjit. *Digital Signal Processing .* Mcgraw Hill.

Nichols, Randall, e Panos Lekkas. *Wireless Security: Models, Threats, and Solutions.* McGraw-Hill, 2002.

Ohrtman, Franklin. *Softswitch Architecture For VoIP.* McGraw-Hill Professional, 2002.

Otto, Stephen Robert, e James Denier. *An Introduction to Programming and Numerical Methods in MATLAB.* Springer , 2005.

Porter, Thomas, et al. *Practical VoIP Security.* Canada: Syngress Publishing,Inc. , 2006.

Whitfield, Diffie, e Martin Hellman. *Multi-user cryptographic techniques.* 1976.

Ferreira, Paulo. *Segurança e Criptografia.* 2008.

http://en.wikipedia.org/wiki/Scrambler, Novembro 2008

http://islab.oregonstate.edu/koc/ece575/02Project/Kie+Raj/, Janeiro 2009

http://www.infowester.com/criptografia.php, Janeiro 2009

http://webscripts.softpedia.com/script/Scientific-Engineering-Ruby/Mathematics/big-modulo-function-34101.html, Fevereiro 2009

http://homepage.smc.edu/morgan_david/vpn/assignments/assgt-primitive-roots.htm, Fevereiro 2009